THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

*PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A*
*Volume 25, Number 3/2024, pp. 225–239*
DOI: 10.59277/PRA-SER.A.25.3.08

# ERROR-FREE TRACKING CONTROL OF DISCRETE-TIME SYSTEMS BY COMBINING DATA-DRIVEN CONTROL WITH MODEL-BASED ILC

Tiantian LU, Guojun LI, Yingsheng FAN, Dongjie CHEN

Zhejiang Police College, Basic Courses Department, Hangzhou, 310053, China
Corresponding author: Guojun LI, E-mail: liguojun@zjjcxy.cn

**Abstract**. For discrete-time linear systems, by applying the advantages of data-driven methods and iterative learning control, this paper proposes three data-driven learning control algorithms based on the linearity of the system. For iterative learning control, if the initial state shift is zero in each iteration and the system matrix is known, the algorithm presented in this paper can ensure that the system achieve complete tracking in the entire interval through a finite number of iterations. If the initial state shift is arbitrary in each iteration and the system matrix is unknown, the algorithm proposed in this paper can ensure that the system achieve complete tracking at all points except the initial point through finite iterations. Even for systems with non-repetitive processes, if the system matrix is unknown, the method in this paper can ensure that the system achieves complete tracking at all points except a finite number of points in the initial stage. Finally, the effectiveness of the three algorithms is verified by three examples, and the superiority of the proposed algorithms is demonstrated by comparison.

***Keywords:*** discrete-time systems, data-driven, iterative learning control, initial state shift.

## 1. INTRODUCTION

Data-driven control originated from the computer field, but has developed rapidly in the control field in the past two decades. The stability of data-driven control systems is essentially related to the model [1] and the data utilized. According to the utilization of data, data-driven control can be roughly divided into three types of control theories and methods, including offline-based, online-based, and offline-online-combined. Among the data-driven control methods based on offline data, the most representative one is the PID (Proportional Integral Differential) control. In the data-driven control methods based on online data, model-free adaptive control has attracted the attention. The combination of offline and online data-driven control methods mainly includes iterative learning control (ILC), repetitive learning control (RLC), and lazy learning control [2].

PID controller is of great concern in the application of offline data control methods. The PID controller consists of three parts: proportional unit, integral unit, and differential unit. Since the mid-20th century, although various advanced controllers have been introduced, the PID controller has become the most commonly used method in industrial control processes due to its simple structure, low model requirements, and ease of operation. In practical applications, the structure of the PID controller is changed based on the control process, control situations, and control requirements, and only some units are taken to construct the controller, such as proportional controller, proportional integral controller, proportional differential controller, etc. As of the year 2000, at least 95% of industrial process control is still PID control [3]. When applying PID controller, the most core content is the tuning of control parameters. Since the Z-N (Ziegler Nichols) method was proposed, many techniques have been used for parameter tuning of PID controllers. In engineering practice, there are many methods that have achieved good results [4]. It is worth mentioning that Astrom and Hagglund have published two monographs on PID controller tuning [5]. In general, PID controllers can enable the system to achieve asymptotic tracking.

The main ways to use online data for control include the following: model-free control based on SPSA (Simultaneous perturbation stochastic approximation) [6–9], unfalsified control (UC) [10–13] and model free

adaptive control (MFAC) [14–18]. Since Spall proposed the SPSA model free control method in 1992 [6], the SPSA method advanced in [7] obtains the gain value of the PID controller by minimizing the objective function. The simulation results show that the PID gains optimized by SPSA can provide better control effect with random uncertainty. [8] improves the SPSA algorithm by combining gradient approximation algorithm. The analysis shows that the estimation sequence generated by the improved algorithm will almost certainly converge to its optimal point. The self-tuning mechanism proposed in [9], combined with the SPSA method, provides effective vibration suppression effects for various controlled objects without manually adjusting the controller. However, the convergence speed of the model free control method based on SPSA needs to be improved. Chen et al. constructed a switching algorithm using the UC algorithm [11], which is proposed by Safonov et al. in 1997 [10], to ensure system stability. In [12], combining $H_\infty$ control and UC, the feasibility of the algorithm is ensured by increasing the degree of freedom of the candidate set of controllers. MFAC was proposed by Hou in 1994 [14], and has been gradually improved after nearly 30 years of development [15, 16]. The basic idea is to replace the original discrete nonlinear model with a dynamic linear model, where only input and output data are used to estimate relevant parameters online, thereby achieving model-free adaptive control. To date, the MFAC method has been widely used [17–19]. It is worth noting that model-free control methods do not exclude the use of system model information. In fact, tracking effect is much better if system information is used. For these online-based control methods, it is generally difficult to achieve complete tracking.

Lazy learning is a supervised machine learning algorithm that was first applied to control by Schaal and Atkeson [20], and then popularized in subsequent development [21–23]. Its main idea is to find a mapping between inputs and outputs based on training data. Yang et al. use the lazy learning algorithm to adjust the parameters of the online T-S fuzzy model and controller, and the simulation results show that this method can significantly improve the tracking accuracy and energy efficiency of high-speed trains [21]. The controller designed by Hou et al., combining lazy learning and MFAC, not only exhibits good robustness, but also achieves adaptive prediction for mutations in expected value [22]. In [23], a controller is proposed for real-time power generation control based on inert reinforcement learning, which can achieve the highest control performance. RLC, which uses information from previous cycles and the current cycle to adjust the current input to improve tracking performance, is generally applicable to systems that run on successive cycles [24]. In contrast, ILC uses information from previous iterations and current iteration to adjust current input to improve tracking performance, and is generally applicable to systems that run repeatedly on a finite interval [25–27]. Because of its simple algorithm and small amount of calculation, ILC is widely used in products assemble line, intelligent robot control, chemical process control and other occasions [28–31]. Both RLC and ILC have one advantage: suppressing repetitive disturbances. Although RLC and ILC can achieve complete tracking when the number of repetitions and iterations approaches infinity, in reality, the number of repetitions or iterations cannot increase indefinitely. Therefore, RLC and ILC can only achieve asymptotic tracking in the actual process.

When applying the ILC method, it is generally required the initial state shift to be zero during each experiment, because the initial value problem not only affects the tracking effect, but also affects the stability of the system [32, 33]. However, due to measurement errors, experimental environment, and other factors, it is difficult to make the initial state shift equal to zero during each experiment or iteration. As a consequence, the ILC problem with initial state shift has been concerned. For discrete-time systems, if there is initial state shift, even if the number of iterations tends to infinity, it is still difficult to achieve complete tracking, which has become an open topic of ILC [34–39].

For a class of discrete-time linear systems with arbitrary initial state shifts in the iterations, this paper applies the data-driven and ILC method to achieve error-free tracking over the whole time interval (except the initial point) after a limited number of iterations. Even for non-repetitive processes, the proposed algorithm can still achieve error-free tracking a specified interval. Finally, the effectiveness of the proposed algorithm is verified by simulation examples.

The main innovations of this work are as follows. (1) The control algorithm proposed in this paper can achieve complete tracking with initial state shift without knowing the system parameters. (2) The data-driven control process fully utilizes the linearity of the system, making the control input design simple and parameter calculation easy. (3) Compared with general iterative learning control, a smaller number of iterations can

achieve better tracking results, that is, error free tracking.

The remaining parts of the paper are composed as follows. Section 2 presents the class of linear systems to be discussed in this paper. In Section 3, a controller is designed for the case where the system matrix $A$ is known and has no initial state shift. Section 4 is about designing controllers for situations where the system matrix $A$ is unknown and there is initial state shift. In Section 5, the controller design scheme for non repetitive processes is discussed. Multiple simulation examples are employed in Section 6, and conclusions are drawn in Section 7.

## 2. PROBLEM FORMULATION

Consider the following linear system

$$\begin{cases} \boldsymbol{x}_k(t+1) = A\boldsymbol{x}_k(t) + B\boldsymbol{u}_k(t), \\ \boldsymbol{y}_k(t) = C\boldsymbol{x}_k(t), \end{cases} \tag{1}$$

where $t = 0,1,2,\cdots,N$; $k = 1,2,\cdots$ denotes the number of iterations; $\boldsymbol{x}_k(t) \in R^n, \boldsymbol{y}_k(t) \in R^r, \boldsymbol{u}_k(t) \in R^m$ represent the system state, control input, and output vectors for the $k-$th iteration, respectively. $A,B,C$ are system parameter matrices of appropriate dimensions.

Let $\boldsymbol{e}_k(t) = \boldsymbol{y}_r(t) - \boldsymbol{y}_k(t)$ represent the $k-$th output error, where $\boldsymbol{y}_r(t)$ is the given reference trajectory, and set $\boldsymbol{x}_r(t)$ to represent the given reference state.

For ILC, there is not much difference between linear systems and nonlinear systems using contraction mapping principle. Actually, for linear systems, utilizing the linearity of the systems in the control process can produce unexpected results. The purpose of this paper is to solve the problem of how to ensure the complete tracking of the system under multiple repetitions with the aid of the data-driven methods and the linearity of the system. In the process of applying data-driven control, if the system matrix $A$ is known, the information of $A$ will be used. However, if the system matrix $A$ is unknown, this paper only needs to use the linearity of the system to achieve error free tracking.

## 3. CONTROLLER DESIGN AND CONVERGENCE ANALYSIS WITH $A$ KNOWN

In this section, the controller design requires not only that the system matrix $A$ is precisely known, but also that the initial state of the system satisfies the following assumption.

*Assumption* 1. The initial state is $\boldsymbol{x}_k(0) = \boldsymbol{x}_r(0)$, that is, the positioning is accurate at each iteration and there is no initial state shift.

### 3.1. Controller design

First, we consider the following learning law

$$\boldsymbol{u}_{k+1}(t) = \boldsymbol{u}_k(t) + \Gamma \boldsymbol{e}_k(t+1). \tag{2}$$

where $\Gamma$ is the gain matrix of appropriate dimension. After repeating the above controller multiple times, we can obtain the corresponding state variables $\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots, \boldsymbol{x}_n(t)$. Applying the system (1), we get

$$\boldsymbol{x}_k(t+1) - A\boldsymbol{x}_k(t) = B\boldsymbol{u}_k(t), \;\; k = 1,2,\cdots,n. \tag{3}$$

Suppose there is an ideal control law $\boldsymbol{u}_{1,d}(t)$, such that

$$\boldsymbol{x}_{1,d}(t+1) - A\boldsymbol{x}_{1,d}(t) = B\boldsymbol{u}_{1,d}(t). \tag{4}$$

Let $\boldsymbol{\xi}_{1,k}(t) = \boldsymbol{x}_k(t+1) - A\boldsymbol{x}_k(t)$ $(k = 1,2,\cdots,n)$, $\boldsymbol{\xi}_{1,d}(t) = \boldsymbol{x}_r(t+1) - A\boldsymbol{x}_r(t)$. If the vectors $\boldsymbol{\xi}_{1,1}(t)$, $\boldsymbol{\xi}_{1,2}(t)$, $\cdots$, $\boldsymbol{\xi}_{1,n}(t)$ are linearly independent, then the following equation must be true.

$$\boldsymbol{\xi}_{1,d}(t) = \alpha_{1,1}(t)\boldsymbol{\xi}_{1,1}(t) + \alpha_{1,2}(t)\boldsymbol{\xi}_{1,2}(t) + \cdots + \alpha_{1,n}(t)\boldsymbol{\xi}_{1,n}(t). \tag{5}$$

Correspondingly, the following ideal controller can be obtained,

$$\boldsymbol{u}_{1,d}(t) = \alpha_{1,1}(t)\boldsymbol{u}_1(t) + \alpha_{1,2}(t)\boldsymbol{u}_2(t) + \cdots + \alpha_{1,n}(t)\boldsymbol{u}_n(t), \tag{6}$$

where the parameters are computed by $(\alpha_{1,1}(t), \alpha_{1,2}(t), \cdots, \alpha_{1,n}(t))^T = (\boldsymbol{\xi}_{1,1}(t)\ \boldsymbol{\xi}_{1,2}(t)\ \cdots\ \boldsymbol{\xi}_{1,n}(t))^{-1}\boldsymbol{\xi}_{1,d}(t)$.

### 3.2. Convergence analysis

Suppose that when control law (6) is applied to system (1), the corresponding state vector $\boldsymbol{x}_{1,d}(t)$, output vector $\boldsymbol{y}_{1,d}(t)$ and tracking error $\boldsymbol{e}_{1,d}(t) = \boldsymbol{y}_r(t) - \boldsymbol{y}_{1,d}(t)$ can be obtained. For system (1) and control law (6), there is the following convergence theorem.

THEOREM 1. *Assuming system* (1) *satisfies Assumption 1, if the system matrix A is accurately known and the state vector is measurable, and the vectors* $\boldsymbol{\xi}_{1,1}(t)$, $\boldsymbol{\xi}_{1,2}(t)$, $\cdots$, $\boldsymbol{\xi}_{1,n}(t)$ *are linearly independent, then the control law* (6) *can stabilize system* (1) *and make the tracking error* $\boldsymbol{e}_{1,d}(t) = 0$, $t \in \{0,1,\cdots,N\}$.

*Proof.* Applying control law (6) to system (1), we get

$$\boldsymbol{x}_r(t+1) - \boldsymbol{x}_{1,d}(t+1) = \boldsymbol{x}_r(t+1) - (A\boldsymbol{x}_{1,d}(t) + B\boldsymbol{u}_{1,d}(t)). \tag{7}$$

Substitute (6) into the above equation to obtain

$$\boldsymbol{x}_r(t+1) - \boldsymbol{x}_{1,d}(t+1) = \boldsymbol{x}_r(t+1) - (A\boldsymbol{x}_{1,d}(t) + B(\alpha_{1,1}(t)\boldsymbol{u}_1(t) + \alpha_{1,2}(t)\boldsymbol{u}_2(t) + \cdots + \alpha_{1,n}(t)\boldsymbol{u}_n(t))). \tag{8}$$

By inserting (4) and (5) into the above equation, we can derive

$$\boldsymbol{x}_r(t+1) - \boldsymbol{x}_{1,d}(t+1) = \boldsymbol{x}_r(t+1) - (A\boldsymbol{x}_{1,d}(t) + \boldsymbol{x}_r(t+1) - A\boldsymbol{x}_r(t)) = A(\boldsymbol{x}_r(t) - \boldsymbol{x}_{1,d}(t)). \tag{9}$$

Then, we have

$$\boldsymbol{e}_{1,d}(t) = \boldsymbol{y}_r(t) - \boldsymbol{y}_{1,d}(t) = C(\boldsymbol{x}_r(t) - \boldsymbol{x}_{1,d}(t)) = CA^t(\boldsymbol{x}_r(0) - \boldsymbol{x}_{1,d}(0)). \tag{10}$$

When $t = 0$, by Assumption 1, i.e. $\boldsymbol{x}_{1,d}(0) = \boldsymbol{x}_r(0)$, we can get $\boldsymbol{e}_{1,d}(t) = 0, t \in \{0,1,\cdots,N\}$. □

*Remark* 1. For system (1) and control law (2),

$$\boldsymbol{x}_{k+1}(t) - \boldsymbol{x}_k(t) = A(\boldsymbol{x}_{k+1}(t-1) - \boldsymbol{x}_k(t-1)) + B\Delta\boldsymbol{u} = A(\boldsymbol{x}_{k+1}(t-1) - \boldsymbol{x}_k(t-1)) + B\Gamma\boldsymbol{e}_k(t). \tag{11}$$

Let $\boldsymbol{\delta}_k(t) = \boldsymbol{x}_r(t) - \boldsymbol{x}_k(t)$, then (11) can be converted into

$$\boldsymbol{\delta}_{k+1}(t) = (I - B\Gamma C)\boldsymbol{\delta}_k(t) + A(\boldsymbol{\delta}_{k+1}(t-1) - \boldsymbol{\delta}_k(t-1)). \tag{12}$$

Set $\boldsymbol{\eta}_k(t) = \boldsymbol{\delta}_{k+1}(t-1) - \boldsymbol{\delta}_k(t-1)$, we have

$$\boldsymbol{\eta}_k(t+1) = \boldsymbol{\delta}_{k+1}(t) - \boldsymbol{\delta}_k(t) = A(\boldsymbol{\delta}_{k+1}(t-1) - \boldsymbol{\delta}_k(t-1)) - B\Gamma C\boldsymbol{\delta}_k(t) = A\boldsymbol{\eta}_k(t) - B\Gamma C\boldsymbol{\delta}_k(t). \tag{13}$$

Then the following model is obtained

$$\begin{pmatrix} \boldsymbol{\eta}_k(t+1) \\ \boldsymbol{\delta}_{k+1}(t) \end{pmatrix} = \begin{pmatrix} A & -B\Gamma C \\ A & I - B\Gamma C \end{pmatrix} \begin{pmatrix} \boldsymbol{\eta}_k(t) \\ \boldsymbol{\delta}_k(t) \end{pmatrix}. \tag{14}$$

It can be seen that (14) takes the 2-D Roesser model structure. Therefore, for causal controller (2), the tracking error of system (1) asymptotically converges to zero if $\rho(A)<1$, $\rho(I-B\Gamma C)<1$ and $\rho(|-A(I-A)^{-1}B\Gamma C+I-B\Gamma C|)<1$. However, the proposed method does not require that $k\to\infty$ to achieve convergence. In other words, unlike classical ILC, the conclusion of this method is $\boldsymbol{e}_{1,d}(t)=0$, rather than the limit being equal to zero.

*Remark* 2. Multiplying the matrix $C$ on both sides of (12), we have

$$\boldsymbol{e}_{k+1}(t)=C\boldsymbol{\delta}_{k+1}(t)=(I-CB\Gamma)\boldsymbol{e}_k(t)+CA(\boldsymbol{\delta}_{k+1}(t-1)-\boldsymbol{\delta}_k(t-1)). \tag{15}$$

It follows that

$$\begin{pmatrix}\boldsymbol{\eta}_k(t+1)\\ \boldsymbol{e}_{k+1}(t)\end{pmatrix}=\begin{pmatrix}A & -B\Gamma\\ CA & I-CB\Gamma\end{pmatrix}\begin{pmatrix}\boldsymbol{\eta}_k(t)\\ \boldsymbol{e}_k(t)\end{pmatrix}. \tag{16}$$

Thereby, if $\rho(A)<1$, $\rho(I-CB\Gamma)<1$ and $\rho(|-CA(I-A)^{-1}B\Gamma+I-CB\Gamma|)<1$, control law (2) can also ensure the convergence of the system (1). Furthermore, the prerequisite for $\rho(I-CB\Gamma)<1$ is that the matrix $CB$ is right invertible (row full rank). In fact, from our proof above, it is clear that the method in this paper does not require $\rho(I-B\Gamma C)<1$ or $\rho(I-CB\Gamma)<1$, but in order to ensure the validity of the control variable, we recommend assuming $\rho(I-B\Gamma C)<1$ or $\rho(I-CB\Gamma)<1$ holds.

*Remark* 3. When the dimension $n$ is large, it is difficult to find linearly independent vectors $\boldsymbol{\xi}_{1,1}(t)$, $\boldsymbol{\xi}_{1,2}(t)$, $\cdots$, $\boldsymbol{\xi}_{1,n}(t)$. At this point, we can reduce the requirement and choose linearly independent vectors $\boldsymbol{\xi}_{1,1}(t)$, $\boldsymbol{\xi}_{1,2}(t)$, $\cdots$, $\boldsymbol{\xi}_{1,s}(t)$ $(s<n)$. Define $\Xi_1=(\boldsymbol{\xi}_{1,2}(t)\cdots\boldsymbol{\xi}_{1,s}(t))$. Obviously, $(\Xi_1^T\Xi_1)^{-1}\Xi_1^T$ is the Moore-Penrose inverse of $\Xi_1$, because it is column full rank. Therefore, the ideal controller can be designed as

$$\boldsymbol{u}_{1,d}(t)=\alpha_{1,1}(t)\boldsymbol{u}_1(t)+\alpha_{1,2}(t)\boldsymbol{u}_2(t)+\cdots+\alpha_{1,s}(t)\boldsymbol{u}_s(t), \tag{17}$$

where$(\alpha_{1,1}(t),\alpha_{1,2}(t),\cdots,\alpha_{1,s}(t))^T=(\Xi_1^T\Xi_1)^{-1}\Xi_1^T\boldsymbol{\xi}_{1,d}(t)$.

## 4. CONTROLLER DESIGN AND CONVERGENCE ANALYSIS WITH *A* UNKNOWN

In this section, the controller design process not only involves the unknown system matrix $A$, but also involves more complex initial state conditions that satisfy the following assumption.

*Assumption* 2. The initial state $\boldsymbol{x}_k(0)$ is random, that is, the positioning is not accurate enough in each iteration and there are initial state shifts.

### 4.1. Controller design

Due to the arbitrary initial state shifts of the system, in order to quickly stabilize the system (1), we first consider the following ILC law with feedback:

$$\boldsymbol{u}_{k+1}(t)=\boldsymbol{u}_k(t)+\Gamma_1\boldsymbol{e}_k(t+1)+\Gamma_2\boldsymbol{e}_{k+1}(t) \tag{18}$$

with $\boldsymbol{u}_0(t)=0$. Similarly, the controller (18) is iterated multiple times to obtain a sequence of iterative state variables $\boldsymbol{x}_{i+1}(t+1)$, $\boldsymbol{x}_{i+2}(t+1)$, $\cdots$, $\boldsymbol{x}_{i+n}(t+1)$ $(i\geq 0)$, and the corresponding control variables are denoted as $\boldsymbol{u}_{i+1}(t)$, $\boldsymbol{u}_{i+2}(t)$, $\cdots$, $\boldsymbol{u}_{i+n}(t)$ $(i\geq 0)$. Similar to Section 3.1, we assume that there exists an ideal control law $\boldsymbol{u}_{2,d}(t)$, such that

$$\boldsymbol{x}_{2,d}(t+1)-A\boldsymbol{x}_{2,d}(t)=B\boldsymbol{u}_{2,d}(t). \tag{19}$$

As the iterative state variable $\boldsymbol{x}_k(t)$ in the system dynamics model (1) is $n-$dimensional, the following

equations are true if the vectors $\boldsymbol{x}_{i+1}(t)$, $\boldsymbol{x}_{i+2}(t)$, $\cdots$, $\boldsymbol{x}_{i+n}(t)$ $(i \geq 0, t \in \{0,1,\cdots,N\})$ are linearly independent.

$$\begin{array}{rl} \boldsymbol{x}_{2,d}(t) = & \beta_1 \boldsymbol{x}_{i+1}(t) + \cdots + \beta_n \boldsymbol{x}_{i+n}(t), \\ \boldsymbol{x}_r(t+1) = & \alpha_{2,1} \boldsymbol{x}_{i+1}(t+1) + \cdots + \alpha_{2,n} \boldsymbol{x}_{i+n}(t+1). \end{array} \tag{20}$$

By inserting (20) into (19), we can derive

$$\alpha_{2,1} \boldsymbol{x}_{i+1}(t+1) + \cdots + \alpha_{2,n} \boldsymbol{x}_{i+n}(t+1) = A(\beta_1 \boldsymbol{x}_{i+1}(t) + \cdots + \beta_n \boldsymbol{x}_{i+n}(t)) + B\boldsymbol{u}_{2,d}(t). \tag{21}$$

If $\alpha_{2,i} = \beta_i (1 \leq i \leq n)$ , then

$$\alpha_{2,1} \boldsymbol{x}_{i+1}(t+1) + \cdots + \alpha_{2,n} \boldsymbol{x}_{i+n}(t+1) = A(\alpha_{2,1} \boldsymbol{x}_{i+1}(t) + \cdots + \alpha_{2,n} \boldsymbol{x}_{i+n}(t)) + B\boldsymbol{u}_{2,d}(t). \tag{22}$$

Because $\boldsymbol{x}_{i+1}(t+1)$, $\boldsymbol{x}_{i+2}(t+1)$, $\cdots$, $\boldsymbol{x}_{i+n}(t+1)$ are generated by the corresponding control variables $\boldsymbol{u}_{i+1}(t)$, $\boldsymbol{u}_{i+2}(t)$, $\cdots$, $\boldsymbol{u}_{i+n}(t)$, we can set $\boldsymbol{u}_{2,d}(t) = \alpha_{2,1}\boldsymbol{u}_{i+1}(t) + \cdots + \alpha_{2,n}\boldsymbol{u}_{i+n}(t)$. But in reality, due to the linear independence of $\boldsymbol{x}_{i+1}(t+1), \boldsymbol{x}_{i+2}(t+1), \cdots, \boldsymbol{x}_{i+n}(t+1)$, and $\boldsymbol{x}_r(t+1) = \alpha_{2,1}\boldsymbol{x}_{i+1}(t+1) + \cdots + \alpha_{2,n}\boldsymbol{x}_{i+n}(t+1)$, we have

$$(\alpha_{2,1}(t), \alpha_{2,2}(t), \cdots, \alpha_{2,n}(t))^T = \chi_i(t+1)^{-1} \boldsymbol{x}_r(t+1), \tag{23}$$

where $\chi_i(t) = (\boldsymbol{x}_{i+1}(t)\ \boldsymbol{x}_{i+2}(t)\ \cdots\ \boldsymbol{x}_{i+n}(t))$. Similarly,

$$(\beta_1(t), \beta_2(t), \cdots, \beta_n(t))^T = \chi_i(t)^{-1} \boldsymbol{x}_{2,d}(t). \tag{24}$$

By the properties of solutions to the linear systems, it is clear that $\alpha_{2,i} = \beta_i (1 \leq i \leq n)$ cannot be guaranteed. To get around this problem, we make the following settings:

$$\left\{ \begin{array}{rcl} \boldsymbol{x}_{2,d}(t) & = & \beta_1 \boldsymbol{x}_{i+1}(t) + \cdots + \beta_{2n} \boldsymbol{x}_{i+2n}(t), \\ \boldsymbol{x}_r(t+1) & = & \alpha_{2,1} \boldsymbol{x}_{i+1}(t+1) + \cdots + \alpha_{2,2n} \boldsymbol{x}_{i+2n}(t+1), \\ 0 & = & \alpha_{2,1}(t) - \beta_1(t), \\ & \vdots & \\ 0 & = & \alpha_{2,2n}(t) - \beta_{2n}(t), \end{array} \right. \tag{25}$$

which can be rewritten as

$$(\boldsymbol{x}_r^T(t+1), \boldsymbol{x}_{2,d}^T(t), 0, \cdots, 0)^T = \Xi_2(t)(\alpha_{2,1}(t), \cdots, \alpha_{2,2n}(t), \beta_1(t), \cdots, \beta_{2n}(t))^T, \tag{26}$$

where

$$\Xi_2(t) = \begin{pmatrix} \Xi_{2,1}(t+1) & \mathbf{0}_{n \times 2n} \\ \mathbf{0}_{n \times 2n} & \Xi_{2,1}(t) \\ \mathbf{I}_{2n \times 2n} & -\mathbf{I}_{2n \times 2n} \end{pmatrix}, \tag{27}$$

with $\Xi_{2,1}(t) = (\boldsymbol{x}_{i+1}(t) \cdots \boldsymbol{x}_{i+2n}(t))$. As a consequence, if $\Xi_2(t)$ is invertible, the ideal controllers can be designed as

$$\boldsymbol{u}_{2,d}(t) = \alpha_{2,1}(t)\boldsymbol{u}_{i+1}(t) + \alpha_{2,2}(t)\boldsymbol{u}_{i+2}(t) + \cdots + \alpha_{2,2n}(t)\boldsymbol{u}_{i+2n}(t). \tag{28}$$

where the parameters are computed by

$$(\alpha_{2,1}(t), \cdots, \alpha_{2,2n}(t), \beta_1(t), \cdots, \beta_{2n}(t))^T = \Xi_2^{-1}(t)(\boldsymbol{x}_r^T(t+1), \boldsymbol{x}_{2,d}^T(t), 0, \cdots, 0)^T. \tag{29}$$

*Remark* 4. Considering that $\boldsymbol{x}_k(t)$ and $\boldsymbol{u}_k(t)$ are $n-$dimensional and $m-$dimensional, respectively, we can execute the control law (18) $n+m$ times to generate a sequence of states $\boldsymbol{x}_1(t)$, $\cdots$, $\boldsymbol{x}_{n+m}(t)$. Next, write the

execution process in matrix form to obtain

$$\begin{pmatrix} A & B \end{pmatrix}\begin{pmatrix} \boldsymbol{x}_1(t) & \boldsymbol{x}_2(t) & \cdots & \boldsymbol{x}_{n+m}(t) \\ \boldsymbol{u}_1(t) & \boldsymbol{u}_2(t) & \cdots & \boldsymbol{u}_{n+m}(t) \end{pmatrix} = (\boldsymbol{x}_1(t+1), \boldsymbol{x}_2(t+1), \cdots, \boldsymbol{x}_{n+m}(t+1)).$$

In general, it is not possible unless $m = 1$. Let $\Omega(t) = \begin{pmatrix} \boldsymbol{x}_1(t) & \boldsymbol{x}_2(t) & \cdots & \boldsymbol{x}_{n+m}(t) \\ \boldsymbol{u}_1(t) & \boldsymbol{u}_2(t) & \cdots & \boldsymbol{u}_{n+m}(t) \end{pmatrix}$, which is a square matrix. In addition, $\Omega(t)$ also can be thought of as a space. Let's assume that $\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots, \boldsymbol{x}_n(t)$ are linearly independent, then the space represented by $(\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots, \boldsymbol{x}_n(t))$ is identical to $(\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots, \boldsymbol{x}_{n+m}(t))$. While $\boldsymbol{u}_k(t)$ is a linear combination of $\boldsymbol{x}_d(t)$ and $\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots, \boldsymbol{x}_{n+m}(t)$, so the dimension of space $\Omega(t)$ is at most $n+1$, and the matrix $\Omega(t)$ is noninvertible when $m > 1$. Therefore, the system matrix $A$ cannot be obtained in this way.

*Remark* 5. Neither $\boldsymbol{u}_1(t), \boldsymbol{u}_2(t), \cdots, \boldsymbol{u}_n(t)$ in the previous section nor $\boldsymbol{u}_{i+1}(t), \boldsymbol{u}_{i+2}(t), \cdots, \boldsymbol{u}_{i+2n}(t)$ in this section, require the control sequence to be continuous along the iteration axis during application. Therefore, in a sense, the form written as $\boldsymbol{u}_{i_1}(t), \boldsymbol{u}_{i_2}(t), \cdots, \boldsymbol{u}_{i_n}(t)$ and $\boldsymbol{u}_{j_1}(t), \boldsymbol{u}_{j_2}(t), \cdots, \boldsymbol{u}_{j_{2n}}(t)$ (where $i_1, i_2, \cdots, i_n$ and $j_1, j_2, \cdots, j_{2n}$ are arbitrary sequences) is more rigorous. The only caveat is that the iterative indices of state $\boldsymbol{x}_k(t)$ and control $\boldsymbol{u}_k(t)$ should match each other.

*Remark* 6. From the above control law design process, it can be seen that when the system matrix $A$ is unknown, the number of iterations required increases significantly, and is closely related to the dimension of the control variable $\boldsymbol{u}_k(t)$. Essentially, it depends on the number of unknowns in the system of linear equations. Only when the number of equations equals the number of unknowns can a system of linear equations have a unique solution. In practice, if $\Xi_2(t)$ is noninvertible (similarly, $\boldsymbol{\xi}_{1,1}, \boldsymbol{\xi}_{1,2}, \cdots, \boldsymbol{\xi}_{1,n}$ in the previous section are linearly dependent) after the corresponding number of iterations, it is necessary to increase the number of iterations until $\Xi_2(t)$ is invertible ($\boldsymbol{\xi}_{1,1}, \boldsymbol{\xi}_{1,2}, \cdots, \boldsymbol{\xi}_{1,n}$ are linearly independent).

### 4.2. Convergence analysis

Suppose that when control law (28) is applied to system (1), the corresponding state vector $\boldsymbol{x}_{2,d}(t)$, output vector $\boldsymbol{y}_{2,d}(t)$ and tracking error $\boldsymbol{e}_{2,d}(t) = \boldsymbol{y}_r(t) - \boldsymbol{y}_{2,d}(t)$ can be obtained. For system (1) and control law (28), there is the following convergence theorem.

THEOREM 2. *Assuming that system* (1) *satisfies Assumption 2, if the system matrix A is unknown while the state vector is measurable, and the matrix* $\Xi_2(t)$ *in the equation* (27) *is invertible, then the control law* (28) *can stabilize system* (1) *and make the tracking error* $\boldsymbol{e}_{2,d}(t) = 0$, $t \in \{1, \cdots, N\}$.

*Proof.* For system (1), by applying the control law (28), we get

$$\begin{aligned} \boldsymbol{\delta}_{2,d}(t+1) = &\ \boldsymbol{x}_r(t+1) - \boldsymbol{x}_{2,d}(t+1) \\ = &\ \boldsymbol{x}_r(t+1) - (A\boldsymbol{x}_{2,d}(t) + B\boldsymbol{u}_{2,d}(t)) \\ = &\ \boldsymbol{x}_r(t+1) - (A\boldsymbol{x}_{2,d}(t) + B(\alpha_{2,1}(t)\boldsymbol{u}_{i+1}(t) + \alpha_{2,2}(t)\boldsymbol{u}_{i+2}(t) + \cdots + \alpha_{2,2n}(t)\boldsymbol{u}_{i+2n}(t))). \end{aligned} \tag{30}$$

Next, combining (25) yields

$$\begin{aligned} \boldsymbol{\delta}_{2,d}(t+1) = &\ \boldsymbol{x}_r(t+1) - (A(\beta_1\boldsymbol{x}_{i+1}(t) + \cdots + \beta_{2n}\boldsymbol{x}_{i+2n}(t)) \\ &+B(\alpha_{2,1}(t)\boldsymbol{u}_{i+1}(t) + \alpha_{2,2}(t)\boldsymbol{u}_{i+2}(t) + \cdots + \alpha_{2,2n}(t)\boldsymbol{u}_{i+2n}(t))). \end{aligned} \tag{31}$$

Because $\beta_i(t) = \alpha_{2,i}(t)$, and based on the dynamic expression of the system (1), the above equation is further converted to

$$\begin{aligned} \boldsymbol{e}_{2,d}(t+1) = &\ C(\boldsymbol{x}_r(t+1) - (\alpha_{2,1}(t)\boldsymbol{x}_{i+1}(t+1) + \alpha_{2,2}(t)\boldsymbol{x}_{i+2}(t+1) + \cdots + \alpha_{2,2n}(t)\boldsymbol{x}_{i+2n}(t+1))) \\ = &\ C(\boldsymbol{x}_r(t+1) - \boldsymbol{x}_r(t+1)) = 0. \end{aligned} \tag{32}$$

Thus, this proof is complete. □

*Remark* 7. From the above control design and proof process, it indicates that if the actual tracked target trajectory (which can be set as $\boldsymbol{x}_r^*(t)$) is not consistent with the iterative target, this method is still effective. It is only necessary to reset the corresponding $\boldsymbol{x}_{2,d}(t+1)$ to $\boldsymbol{x}_r^*(t+1)$ when calculating parameter $\alpha_{2,i}(t)$ in (29).

*Remark* 8. As is well known, for ILC, when designing control laws, it is necessary to consider the initial state value of the system, as the initial state value not only affects the convergence speed of the system, but also affects its stability. However, from the control design process and convergence proof in this section, it is clear that the initial state value of the system has no real impact on the convergence speed and stability of the system, which reflects the robustness of the proposed control method.

## 5. NON-REPETITIVE PROCESS CONTROLLER DESIGN AND CONVERGENCE ANALYSIS

This section specifically explores how to design the controller so that the system can achieve complete tracking within a specified interval when the control process is not repeatable and the system matrix $A$ is unknown. In addition, initial state condition satisfies the following assumption.

*Assumption* 3. The initial state $\boldsymbol{x}(0)$ is random, that is, the positioning is not accurate enough and there are initial state shifts.

### 5.1. Problem formulation

Consider the following linear system

$$\begin{cases} \boldsymbol{x}(t+1) = A\boldsymbol{x}(t) + B\boldsymbol{u}(t), \\ \boldsymbol{y}(t) = C\boldsymbol{x}(t), \end{cases} \tag{33}$$

where $t = 0,1,2,\cdots,N$; $\boldsymbol{x}(t) \in R^n$, $\boldsymbol{y}(t) \in R^r$, $\boldsymbol{u}(t) \in R^m$ represent the system state, control input, and output vectors, respectively. $A, B, C$ are the system parameter matrices with appropriate dimensions.

Let $\boldsymbol{e}(t) = \boldsymbol{y}_r(t) - \boldsymbol{y}(t)$ represent the output error, where $\boldsymbol{y}_r(t)$ is the given reference trajectory, and accordingly set $\boldsymbol{x}_r(t)$ as the given reference state.

### 5.2. Controller design

In this subsection, we divide the controller design into two steps. First, we consider the following control law

$$\boldsymbol{u}(t) = \Gamma \boldsymbol{e}(t). \tag{34}$$

After the control law (34) iteratively runs $i+2n$ steps, we can get the corresponding $\boldsymbol{x}(i+1)$, $\boldsymbol{x}(i+2)$, $\cdots$, $\boldsymbol{x}(i+2n)$. Similarly, we assume that the control law (34) has an ideal control law $\boldsymbol{u}_{3,d}(t)$ after iteratively running $i+2n$ steps, such that

$$\boldsymbol{x}_r(t+1) - A\boldsymbol{x}_r(t) = B\boldsymbol{u}_{3,d}(t), \tag{35}$$

where $t > i+2n$, and

$$\begin{cases} \boldsymbol{x}(t) &= \gamma_1 \boldsymbol{x}(i) + \cdots + \gamma_{2n} \boldsymbol{x}(i+2n-1), \\ \boldsymbol{x}_r(t+1) &= \alpha_{3,1} \boldsymbol{x}(i+1) + \cdots + \alpha_{3,2n} \boldsymbol{x}(i+2n), \\ 0 &= \alpha_{3,1}(t) - \gamma_1(t), \\ &\vdots \\ 0 &= \alpha_{3,2n}(t) - \gamma_{2n}(t). \end{cases} \tag{36}$$

Rewrite the above equation into the following matrix form:

$$(\boldsymbol{x}_r^T(t+1),\boldsymbol{x}^T(t),0,\cdots,0)^T=\Xi_3(i)(\alpha_{3,1}(t),\cdots\alpha_{3,2n}(t),\gamma_1(t),\cdots,\gamma_{2n}(t))^T, \tag{37}$$

where

$$\Xi_3(i)=\begin{pmatrix}\Xi_{3,1}(i+1) & \mathbf{0}_{n\times 2n}\\ \mathbf{0}_{n\times 2n} & \Xi_{3,1}(i)\\ \mathbf{I}_{2n\times 2n} & -\mathbf{I}_{2n\times 2n}\end{pmatrix}, \tag{38}$$

and $\Xi_{3,1}(i)=(\boldsymbol{x}(i)\cdots\boldsymbol{x}(i+2n-1))$. Moreover, if the matrix $\Xi_3(i)$ is invertible, the ideal controllers can be designed as follow,

$$\boldsymbol{u}_{3,d}(t)=\alpha_{3,1}(t)\boldsymbol{u}(i+1)+\alpha_{3,2}(t)\boldsymbol{u}(i+2)+\cdots+\alpha_{3,2n}(t)\boldsymbol{u}(i+2n), \tag{39}$$

with the parameters are computed by

$$(\alpha_{3,1}(t),\cdots\alpha_{3,2n}(t),\gamma_1(t),\cdots,\gamma_{2n}(t))^T=\Xi_3^{-1}(i)(\boldsymbol{x}_r^T(t+1),\boldsymbol{x}^T(t),0,\cdots,0)^T. \tag{40}$$

As a consequence, for the whole control process, we propose the following segmentation controller:

$$\boldsymbol{u}(t)=\begin{cases}\Gamma\boldsymbol{e}(t), & t\le i+2n,\\ \alpha_{3,1}(t)\boldsymbol{u}(i+1)+\alpha_{3,2}(t)\boldsymbol{u}(i+2)+\cdots+\alpha_{3,2n}(t)\boldsymbol{u}(i+2n), & t>i+2n.\end{cases} \tag{41}$$

*Remark* 9. In fact, $\Xi_{3,1}(i)$ in (38) can be designed to vary with time. However, considering issues such as computational complexity and timeliness, we do not choose $\Xi_{3,1}(i)$ to vary with time. But no matter which design method is adopted, it is necessary to ensure that the matrix $\Xi_{3,1}(i)$ is invertible.

### 5.3. Convergence analysis

For the system (33) and the control law (41), there is the following convergence theorem.

THEOREM 3. *Assuming that system* (33) *satisfies Assumption 3, if the system matrix A is unknown but the state vector is measurable,* $\rho(A-B\Gamma C)<1$ *holds, and* $\Xi_3(i)$ *is an invertible matrix, then the control law* (41) *can stabilize the system* (33)*, and ensure that the tracking error* $\boldsymbol{e}(t)$ *asymptotically converges on* $\{0,1,\cdots,i+2n\}$*, and* $\boldsymbol{e}(t)=0$ *on* $\{i+2n+1,\cdots,N\}$.

*Proof.* When $t\ge 1$, substituting the controller (34) into the system (33) yields

$$\begin{aligned}&\boldsymbol{x}(t+1)\\ =\;&A\boldsymbol{x}(t)+B\boldsymbol{u}(t)=A\boldsymbol{x}(t)+B\Gamma\boldsymbol{e}(t)\\ =\;&A\boldsymbol{x}(t)+B\Gamma C(\boldsymbol{x}_r(t)-\boldsymbol{x}(t))\\ =\;&(A-B\Gamma C)\boldsymbol{x}(t)+B\Gamma C\boldsymbol{x}_r(t).\end{aligned} \tag{42}$$

In the above equation, if $\rho(A-B\Gamma C)<1$, then the control law (34) can ensure that the system (33) asymptotically converges on $\{0,1,\cdots,i+2n\}$. The subsequent proof is similar to the proof of Theorem 2 and is omitted here. □

*Remark* 10. For the traditional process control, it is generally only possible to achieve asymptotic tracking and difficult to achieve complete tracking. In contrast, the method provided in this paper can achieve complete tracking a specified interval, which demonstrates the effectiveness of the proposed method.

*Remark* 11. The above control design and proof process indicates that, after finite iterations, the proposed method can make the system (1) achieve complete tracking on $\{1,\cdots,N\}$; Without iteration, the method can still make the system (33) achieve complete tracking on $\{i+2n+1,\cdots,N\}$. Superficially, there is not

significantly difference in the control effect of the proposed method between on repetitive and non-repetitive processes, but this is not the case in reality. In fact, for repetitive processes, this method can suppress repetitive disturbances due to its repeatability, while for non-repetitive processes, this method cannot suppress process disturbances. As for the disturbance problem during the control process, we will conduct specific research in the future.

## 6. NUMERICAL SIMULATION

In this section, we will verify the effectiveness of the proposed algorithms by three examples. Example 1 is the case where the system matrix $A$ is accurately known, Example 2 is for the case where the system matrix $A$ is unknown, and Example 3 is the case of non-repetitive system.

### 6.1. Example 1

In this example, the parameters in the system (1) are as follows

$$A = \begin{bmatrix} 0 & -0.5996 \\ 0.7843 & 0.5994 \end{bmatrix}, \ B = \begin{bmatrix} -0.5965 & 0 & 0.4213 \\ 0.67 & 1.0780 & 0.5657 \end{bmatrix}, \ C = \begin{bmatrix} 0 & 1 \\ 0.5 & 0.3 \\ 0.2 & 0.76 \end{bmatrix}. \tag{43}$$

The operating interval of the system is $t \in \{0, \cdots, 400\}$, and the system reference state is described by $\boldsymbol{x}_r(t) = (10\sin(\frac{\pi t}{200}), 20\cos(\frac{\pi t}{200}))^T$. Correspondingly, the reference trajectory $\boldsymbol{y}_r(t) = C\boldsymbol{x}_r(t)$. The initial state of the system is set to $\boldsymbol{x}_k(0) = (0;20)$. We select $\Gamma = 0.5848$ in the control law (2), then $\rho(I - B\Gamma C) = 0.8604 < 1$. During the simulation process, 2 iterations are performed, and the simulation results are shown in Figs. 1–4.

In Fig. 1, the blue chain line $y_r(1,t)$ and solid line $y_r(2,t)$ represent the two expected outputs, while the red chain line $y_{1,d}(1,t)$ and solid line $y_{2,d}(2,t)$ represent the actual outputs. It is easy to see from the figure that the red and blue curves basically coincide, which reflects the effectiveness of the algorithm in this paper. This can also be confirmed by the error information in Fig. 2, where the red and blue solid lines represent two output errors. The tracking error at each moment is on the order of $10^{-13}$, which is caused by the inevitable error generated by *Matlab* software when calculating the inverse matrix. Theoretically, without calculation errors, the error will be zero at all points.

In Fig. 3, the green dotted, chain, and solid lines represent the inputs during the second iteration (the error during the first iteration is $\boldsymbol{y}_d(t)$), correspondingly, the blue curves represent the inputs at the third iteration, while the red curves represent the actual inputs. Fig. 3 shows that the inputs at the second and third iterations have slight flutters, while the red input signal is smooth. According to the expression of the system (1), if the reference signal is smooth, then the input signal must be smooth. Therefore, it is the slight flutters of the initial stage of the second and third iterations that lead to the slight flutters of the parameters $\alpha_{1,1}(t)$ and $\alpha_{1,2}(t)$ in Fig. 4.

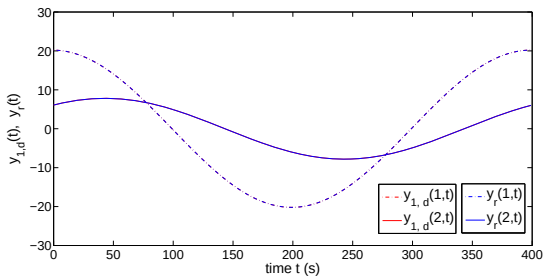


Fig. 1 – Actual output $\boldsymbol{y}_{1,d}(t)$ and reference signal $\boldsymbol{y}_r(t)$.

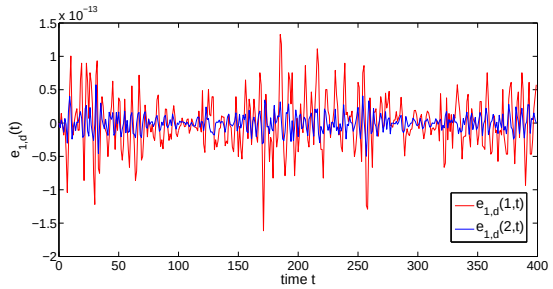


Fig. 2 – Error $\boldsymbol{e}_{1,d}(t)$ .

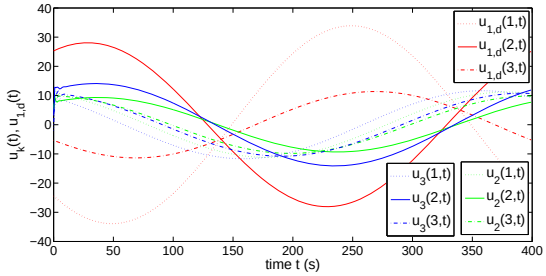


Fig. 3 – Control signals $\boldsymbol{u}_2(t)$, $\boldsymbol{u}_3(t)$ and $\boldsymbol{u}_{1,d}(t)$.

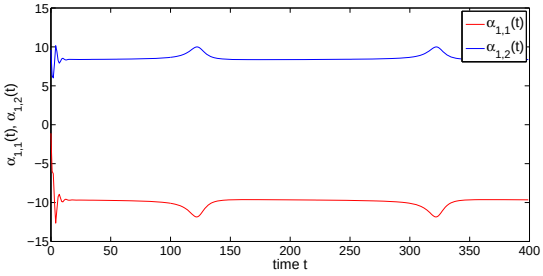


Fig. 4 – Parameters $\alpha_{1,i}(t)(i=1,2)$.

### 6.2. Example 2

In this simulation, we set the parameters in the system (1) as follows

$$A=\begin{bmatrix} 0.1490 & 0.78 & -0.3453 \\ -0.71 & 0.902 & 0.1212 \\ 1.504 & 0.1976 & 0.2534 \end{bmatrix}, B=\begin{bmatrix} 0.1456 & 0.843 & 0.843 \\ 0.2578 & 0.239 & 0.2578 \\ -0.25 & 0.24 & 0.239 \end{bmatrix}, C=\begin{bmatrix} -0.234 & 0.789 & 1.534 \\ 0.478 & 0.982 & 0.76 \\ -0.789 & 0.478 & -0.982 \end{bmatrix} \quad (44)$$

The operating interval of the system is $t \in \{0,\cdots,100\}$, and the system reference state is formulated by $\boldsymbol{x}_r(t)=(20-20\cos(0.02\pi t), 6(\frac{t}{50})^5-15(\frac{t}{50})^4+10(\frac{t}{50})^3, \exp(t/50))^T$. The initial state of the system is given by $\boldsymbol{x}_k(0)=[\text{rand; rand; rand}]$ (rand generates a random number between 0 and 1). We choose

$$\Gamma_2=\begin{bmatrix} -1.9959 & 1.2684 & 0.4103 \\ -28.5193 & -6.4753 & -20.6746 \\ 28.2330 & 6.9730 & 20.1727 \end{bmatrix}, \Gamma_1=\begin{bmatrix} 2.8052 & -3.6351 & 2.7584 \\ 0.4676 & -33.0223 & -0.2747 \\ -1.3468 & 34.5817 & 0.2781 \end{bmatrix} \text{ in control law (2),}$$

thus $\rho(I-B\Gamma C)=0.30<1$. 10 iterations are conducted in the simulation process, and the simulation results are shown in Figs. 5–8.

In Fig. 5, the chain lines represent reference signals $\boldsymbol{y}_r(t)$, while the solid lines represent the actual outputs $\boldsymbol{y}_{2,d}(t)$. Notice that in the figure, the actual outputs and reference signals basically coincide (except for the initial time), which demonstrates the effectiveness of the proposed algorithm. This can also be verified by the error information in the right figure of Fig. 6, where the solid lines represent the output errors, and the tracking error at every moment except the initial moment is $10^{-9}$ level. Similarly, if there is no calculation error, then the error of all points will be zero. This implies that the proposed algorithm is robust to the initial state shifts of the system. The left figure of Fig. 6 shows the error information after the ILC law running 10 times. On account of the influence of arbitrary initial state shifts, though the control law is iterated 10 times, complete tracking was not achieved. And in the initial stage of tracking, the tracking error exhibits fluttering. This indicates that any initial state shift has an impact on tracking performance.

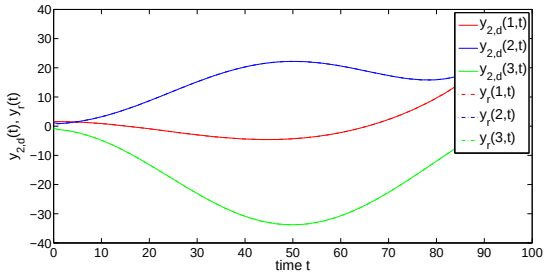


Fig. 5 – Actual output $\boldsymbol{y}_{2,d}(t)$ and reference signal $\boldsymbol{y}_r(t)$.

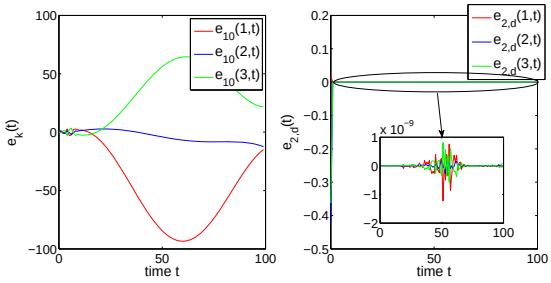


Fig. 6 – Error $\boldsymbol{e}_{10}(t)$ and $\boldsymbol{e}_{2,d}(t)$.

In Fig. 7, the solid lines represent parameters $\alpha_{2,i}(t)(i=1,2,\cdots,6)$, while the chain lines represent parameters $\beta_i(t)(i=1,2,\cdots,6)$. It is precisely because of the flutters of the tracking errors that the flutters of $\alpha_{2,i}(t)$ and $\beta_i(t)$ are caused, which is evident in the left of Fig. 6. But during post-tracking process, the flutters of

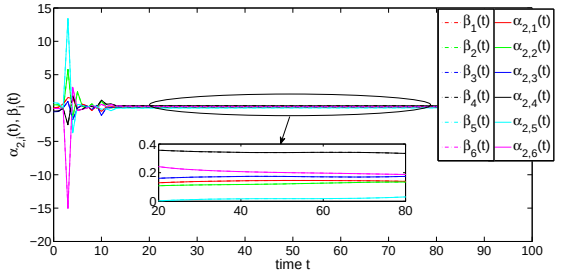


Fig. 7 – Parameters $\alpha_{2,i}(t)$ and $\beta_i(t)$.

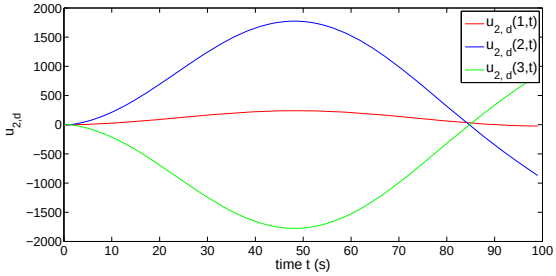


Fig. 8 – Control signal $\boldsymbol{u}_{2,d}(t)$ .

$\alpha_{2,i}(t)$ and $\beta_i(t)$ disappear, reaching a steady state. All in all, Fig. 8 shows that although the parameters $\alpha_{2,i}(t)$ and $\beta_i(t)$ flutter at the beginning, they can still ensure the smoothness of the control signal.

### 6.3. Example 3

In this example, consider the parameters of the system (33) are

$$A=\begin{bmatrix} 1.1 & 0.12 & -0.3 & 0.4 \\ 0.18 & 0.16 & 0.1 & 1.3 \\ 1.5 & 0.1 & 0.2 & 0.9 \\ 0.7 & 1.7 & 0.2 & 0.6 \end{bmatrix},\ B=\begin{bmatrix} 1.14 & 0.42 & 0.843 & -0.5965 \\ 0.0067 & -1.08 & 0.2578 & 3 \\ -0.25 & 0.24 & 0.239 & 0.3421 \\ 0.4213 & 0.67 & 0.078 & 0.5657 \end{bmatrix}, \tag{45}$$

$$C=\begin{bmatrix} 1 & 0 & 1.5 & 0.33 \\ 2 & 0.02 & 1.76 & 0.78 \\ -0.789 & 0.0478 & -1.982 & 0.6721 \\ 0.3402 & 1.3 & 1.2 & 1.76 \end{bmatrix}.$$

The operating interval of the system is $t\in\{0,\cdots,60\}$, and the system reference state is defined as $\boldsymbol{x}_r(t)=(x_r(1,t),\sin(\pi t/30),\cos(\pi t/30),\exp(t/30))^T$, where, $x_r(1,t)=2(6(\frac{t}{30})^5-15(\frac{t}{30})^4+10(\frac{t}{30})^3)$, if $0\le t<30$, $x_r(1,t)=2(6(\frac{60-t}{30})^5-15(\frac{60-t}{30})^4+10(\frac{60-t}{30})^3)$, if $30\le t\le 60$. The initial state of the system is $\boldsymbol{x}_k(0)=5[\text{rand; rand; rand; rand}]$. In the control law (41), we select $i=1$ ($i=0$ if the initial state is arbitrary) and $\Gamma=0.5$, then $\rho(A-B\Gamma C)<1$, thus the control law (41) can ensure that the system (33) tracks asymptotically on $\{0,\cdots,9\}$. Meanwhile, we compare our method with the traditional $P-$type control law $\boldsymbol{u}(t)=\Gamma\boldsymbol{e}(t)$. The comparison results are shown in Figs. 9–12.

In Fig. 9, the red, blue and black solid lines represent the outputs of the proposed method, the output of the traditional $P-$type control law, and the reference trajectory, respectively. In Figs. 10 and 11, the blue solid lines represent the output error and control variable of the proposed method, while the red solid lines represent the output error and control variable of traditional p-type control, respectively. From Fig. 10, it is clear that the control variables of the two control methods are exactly the same on $\{0,\cdots,9\}$, which determines that the outputs of the two methods in Fig. 10 are completely identical, and correspondingly, the error curves in Fig. 10 are completely consistent. But on $\{10,\cdots,60\}$, the difference between the two methods is very obvious. As Fig. 9 shows, the output curve of the method in this article basically coincides with the reference trajectory, and the error curve in Fig. 10 also confirms this point. In comparison, the control effect of the traditional $P-$type control law is significantly worse, which also reflects the superiority of the proposed method.

Unlike the previous two examples, the curves of parameters $\alpha_{3,i}(t)$ and $\gamma_i(t)$ in this example do not exhibit flutter (see Fig. 12). Because $\Xi_3(i)$ in the equation (37) is a constant, correspondingly, $\Xi_3^{-1}(i)$ is also a constant, and the vector $(\boldsymbol{x}_r^T(t+1),\boldsymbol{x}^T(t),0,\cdots,0)^T$ is smooth, so the curves of parameters $\alpha_{3,i}(t)$ and $\gamma_i(t)$ must be smooth.

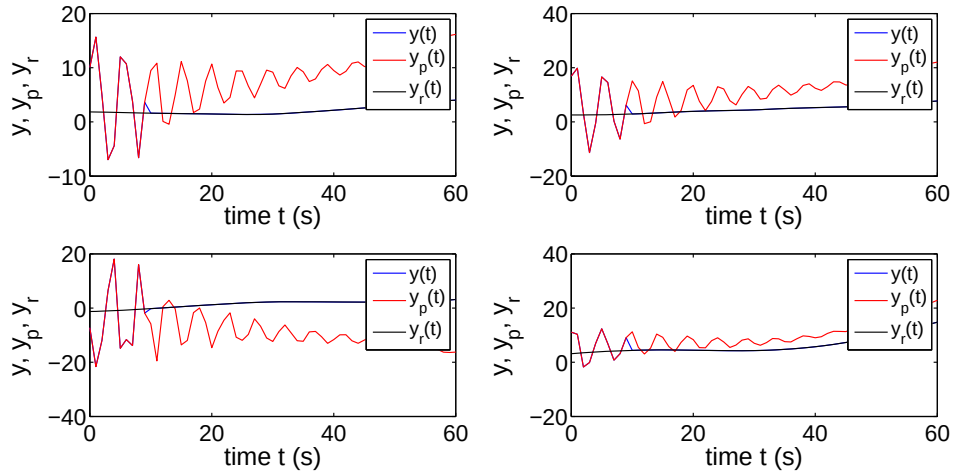


Fig. 9 – Actual outputs $\boldsymbol{y}(t)$, $\boldsymbol{y}_p(t)$ and reference signal $\boldsymbol{y}_r(t)$ .

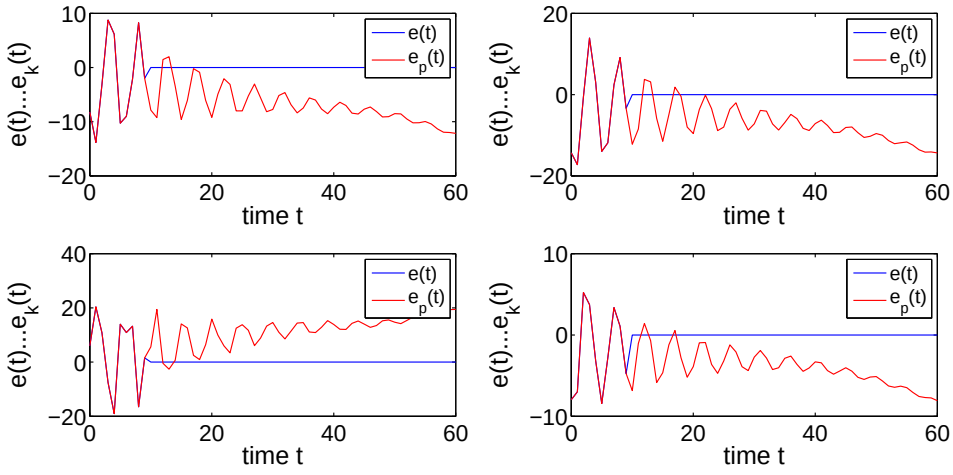


Fig. 10 – Errors $\boldsymbol{e}(t)$ and $\boldsymbol{e}_p(t)$.

## 7. CONCLUSION

This paper investigates the convergence performance of data-driven learning control for the discrete-time linear systems. When discussing convergence, it is divided into three types of problems for discussion. Regardless of the type of system, the linearity of the system is connected to the controller design. The analysis shows that for ILC, the proposed method can achieve complete tracking with finite iterations; In terms of process control, this method can achieve complete tracking a specified interval. The final simulation examples verify the effectiveness of the algorithm and demonstrate its superiority through comparison.

ACKNOWLEDGEMENTS

This work was supported by Zhejiang Police College (2022QTY001).

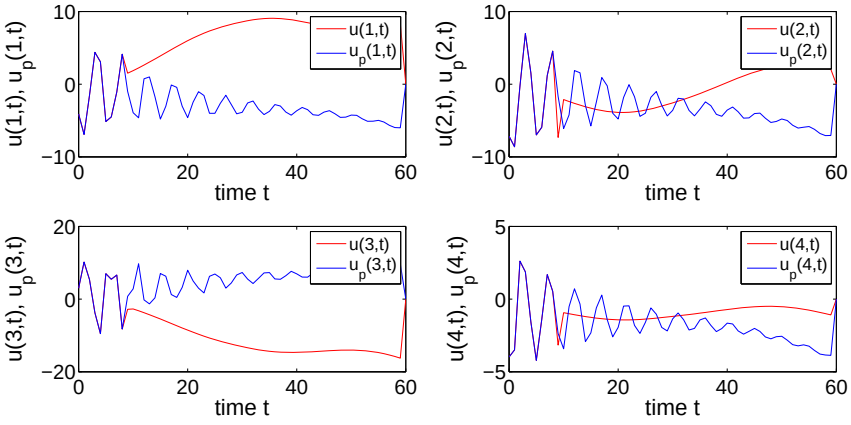


Fig. 11 – Control signals $\boldsymbol{u}(t)$ and $\boldsymbol{u}_p(t)$ .

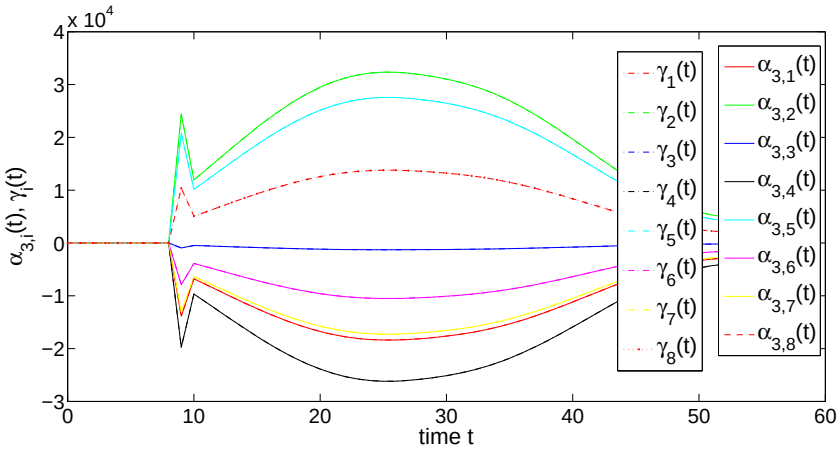


Fig. 12 – Parameters $\alpha_{3,i}(t)$ and $\gamma_i(t)$.

## REFERENCES

[1] Precup R-E, Roman R-C, Safaei A. Data-driven model-free controllers. CRC Press; 2021.

[2] Hou Z-S, Xu J-X. On data-driven control theory: the state of the art and perspective. Acta Automatica Sinica 2009;35(6):650–667.

[3] Wang W, Zhang J-T, Chai T-Y. A survey of advanced PID parameter tuning method. Acta Automatica Sinica 2000;26(3):347–355.

[4] Zhang Y-J, Niu H, Chai T-Y. Data-based un-modeled dynamics driven nonlinear PID. Control Theory & Applications 2020;37:481–491.

[5] Astrom KJ, Hgglund T. PID controllers: theory, design and tuning. Research Triangle Park, North Carolina: Instrument society of America; 1995.

[6] Spall JC. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Transactions on Automatic Control 1992;37(3):332–341.

[7] Singh R, Bhushan B. Adaptive control using stochastic approach for unknown but bounded disturbances and its application in balancing control. Asian Journal of Control 2022;24(3):1304–1320.

[8] Wang L, Spall JC. Improved SPSA using complex variables with applications in optimal control problems. In: 2021 American Control Conference (ACC). 2021, pp. 3519–3524.

[9] Yonezawa A, Yonezawa H, Kajiwara I. Vibration control for various structures with time-varying properties via model-free adaptive controller based on virtual controlled object and SPSA. Mechanical Systems and Signal Processing 2022;170:108801–108813.

[10] Safonov MG, Tsao T-C. The unfalsified control concept and learning. IEEE Transactions on Automatic Control 1997;42(6):843–847.

[11] Chen K, Li S. Unfalsified adaptive PID control for time-varying systems using a fading memory cost function. Circuits, Systems, and Signal Processing 2016;35(9):3172–3191.

[12] Peña RS, Colmegna P, Bianchi FD. Unfalsified control based on the controller parameterisation. International Journal of Systems Science 2015;46(15):2820–2831.

[13] Mishra VK, Markovsky I. The set of linear time-invariant unfalsified models with bounded complexity is affine. IEEE Transactions on Automatic Control 2021;66(9):4432–4435.

[14] Hou Z-S. The parameter identification, adaptive control and model free learning adaptive control for nonlinear systems. Northeastern University; 1994.

[15] Hou Z-S, Xiong S-S. On model-free adaptive control and its stability analysis. IEEE Transactions on Automatic Control 2019;64(11):4555–4569.

[16] Chi R-H, Hui Y, Zhang S-H, Huang B, Hou Z. Discrete-time extended state observer-based model-free adaptive control via local dynamic linearization. IEEE Transactions on Industrial Electronics 2020;67(10):8691–8701.

[17] Li D, Hou Z-S. Perimeter control of urban traffic networks based on model-free adaptive control. IEEE Transactions on Intelligent Transportation Systems 2020;22(10):6460–6472.

[18] Lei T, Hou Z-S, Ren Y. Data-driven model free adaptive perimeter control for multi-region urban traffic networks with route choice. IEEE Transactions on Intelligent Transportation Systems 2020;21(7):2894–2905.

[19] Yu W, Bu X-H, Hou Z-S. Security data-driven control for nonlinear systems subject to deception and false data injection attacks. IEEE Transactions on Network Science and Engineering 2022;9(4):2910–2921.

[20] Schaal S, Atkeson CG. Robot juggling: implementation of memory-based learning. IEEE control systems 1994;14(1):57–71.

[21] Yang H, Zhang K-P, Liu H-E. Online regulation of high speed train trajectory control based on T-S fuzzy bilinear model. IEEE Transactions on Intelligent Transportation Systems 2016;17(6):1496–1508.

[22] Hou Z-S, Liu S-D, Tian T-T. Lazy-learning-based data-driven model-free adaptive predictive control for a class of discrete-time nonlinear systems. IEEE Transactions on Neural Networks and Learning Systems 2017;28(8):1914–1928.

[23] Yin L-F, Li S-Y, Liu H. Lazy reinforcement learning for real-time generation control of parallel cyber-physical-social energy systems. Engineering Applications of Artificial Intelligence 2020;88(1):103380.

[24] Chen Q, Yu X-Q. Adaptive repetitive learning control for a class of nonparametric uncertain systems. Control Theorey & Applications 2020;37(6):1349–1357.

[25] Arimoto S, Kawamura S, Miyazaki F, Bettering operation of robots by learning. Journal of Robotic Systems 1984;2(1):123–140.

[26] Bristow DA, Tharayil M, Alleyne AG. A survey of iterative learning control. IEEE Control Systems Magazine 2006;3(26):96–114.

[27] Xu J-X, Hou Z-S. On learning control: the state of the art and perspective. Acta Automatica Sinica 2005;6(31):943–952.

[28] Jin X. Fault-tolerant iterative learning control for mobile robots non-repetitive trajectory tracking with output constraints. Automatica 2018;94:63–71.

[29] Bouakrif F, Zasadzinski M. Trajectory tracking control for perturbed robot manipulators iterative learning method. International Journal of Advanced Manufacturing Technology 2016;12(3):211–220.

[30] Liu T, Wang X-Z, Chen J-H. Robust PID based indirect-type iterative learning control for batch processes with time-varying uncertainties. Journal of Process Control 2014;24(12):95–106.

[31] Jeong G-M, Choi C-H. Iterative learning control for linear discrete time nonminimum phase systems. Automatica 2002;38:287–291.

[32] Lee H-S, Bien ZZ. Study on robustness of iterative learning control with non-zero initial error. International Journal of Control 1996;64(3):345–359.

[33] Heinzinger G, Fenwick D. Stability of learning control with disturbances and uncertain initial conditions. IEEE Transactions on Automatic Control 1992;37(1):110–114.

[34] Li G-J, Chen D-J, Han Y-S, et al., Iterative learning control with arbitrary initial states for nonlinear systems. Acta Mathematicae Applicatae Sinica 2019;42(4):455–469.

[35] Chien C-J, Hsu C-T, Yao C-Y. Fuzzy system-based adaptive iterative learning control for nonlinear plants with initial state errors. IEEE Transactions on Fuzzy Systems 2004;5(12):724–732.

[36] Zhang C-L, Li J-M. Adaptive iterative learning control for nonlinear pure-feedback systems with initial state error based on fuzzy approximation. Journal of the Franklin Institute 2014;351(3):1483–1500.

[37] Chi R-H, Hou Z-S, Xu J-X. Adaptive ILC for a class of discrete-time systems with iteration-varying trajectory and random initial condition. Automatica 2008;44:2207–2213.

[38] Li X-D, Chow TWS, Ho JKL, Zhang J. Iterative learning control with initial rectifying action for nonlinear continuous systems. IET Control Theory and Applications 2009;3(1):49–55.

[39] Meng D-Y, Jia Y-M, Du J-P, Yuan S. Robust discrete-time iterative learning control for nonlinear systems with varying initial state shifts. IEEE Transactions on Automatic Control 2009;54(11):2626–2631.

*Received March 7, 2024*